AVL's

Pablo Castro
UNRC-Algoritmos I

AVL's

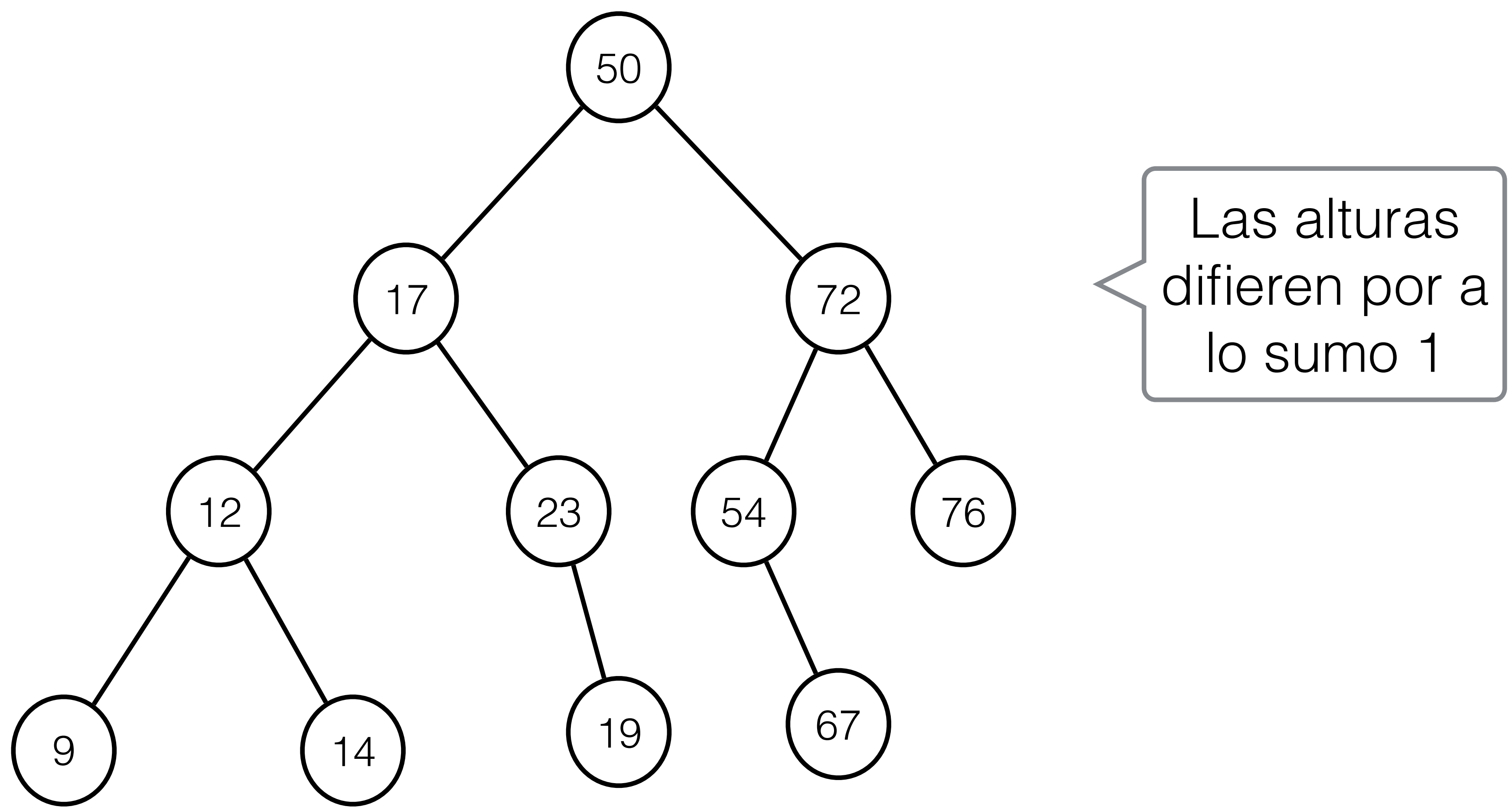
Los AVL's solucionan algunos de los problemas de los ABB's.

- AVL = Adelson-Velskii y Landis (los autores),
- Son ABB's pero con ciertas restricciones que aseguran los árboles se mantienen balanceados:
  - La altura de sus hijos puede diferir en a los sumo 1
- Insertar, eliminar y buscar son O(log n).

Pedir que sean exactamente balanceados no tiene sentido ya que no podríamos insertar ni eliminar.

Un Ejemplo

Un ejemplo de AVL:



Implementando AVL's

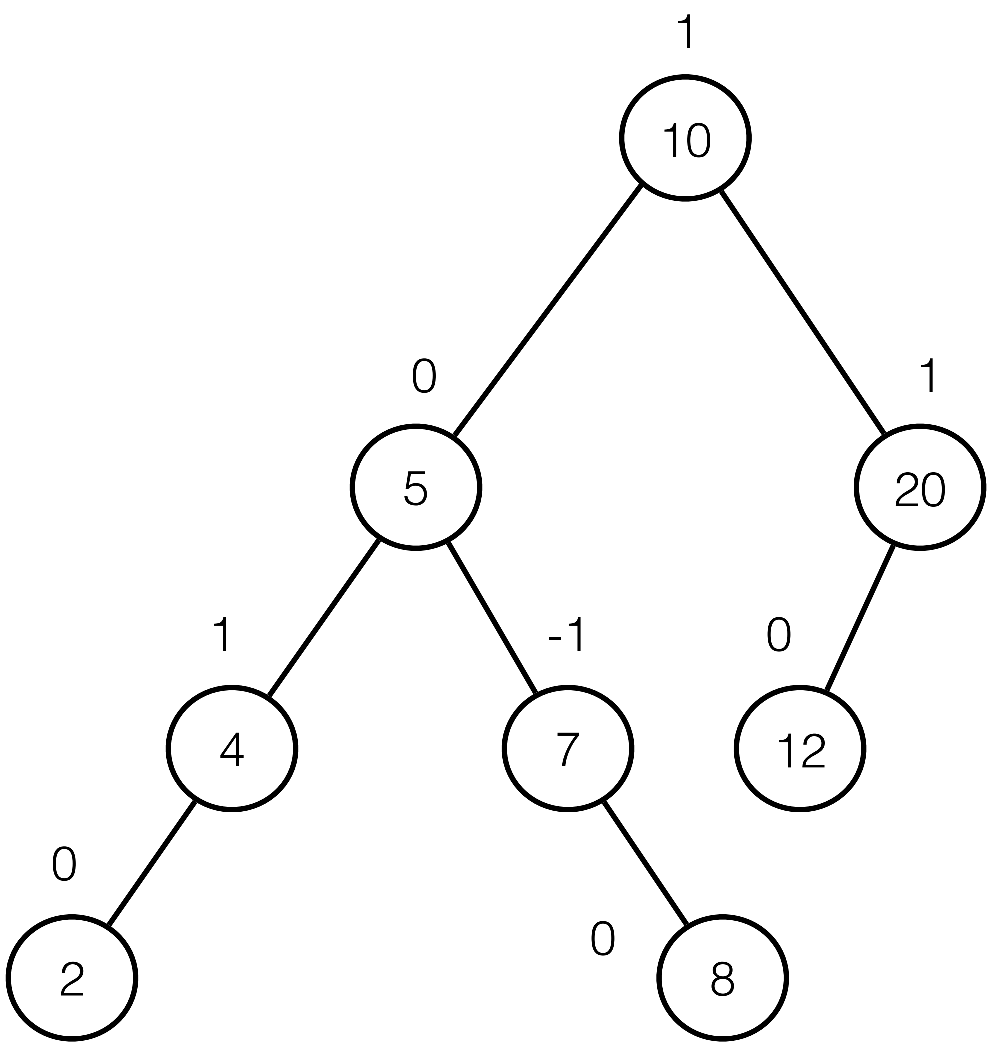
Podemos implementar los AVL's de la siguiente forma:

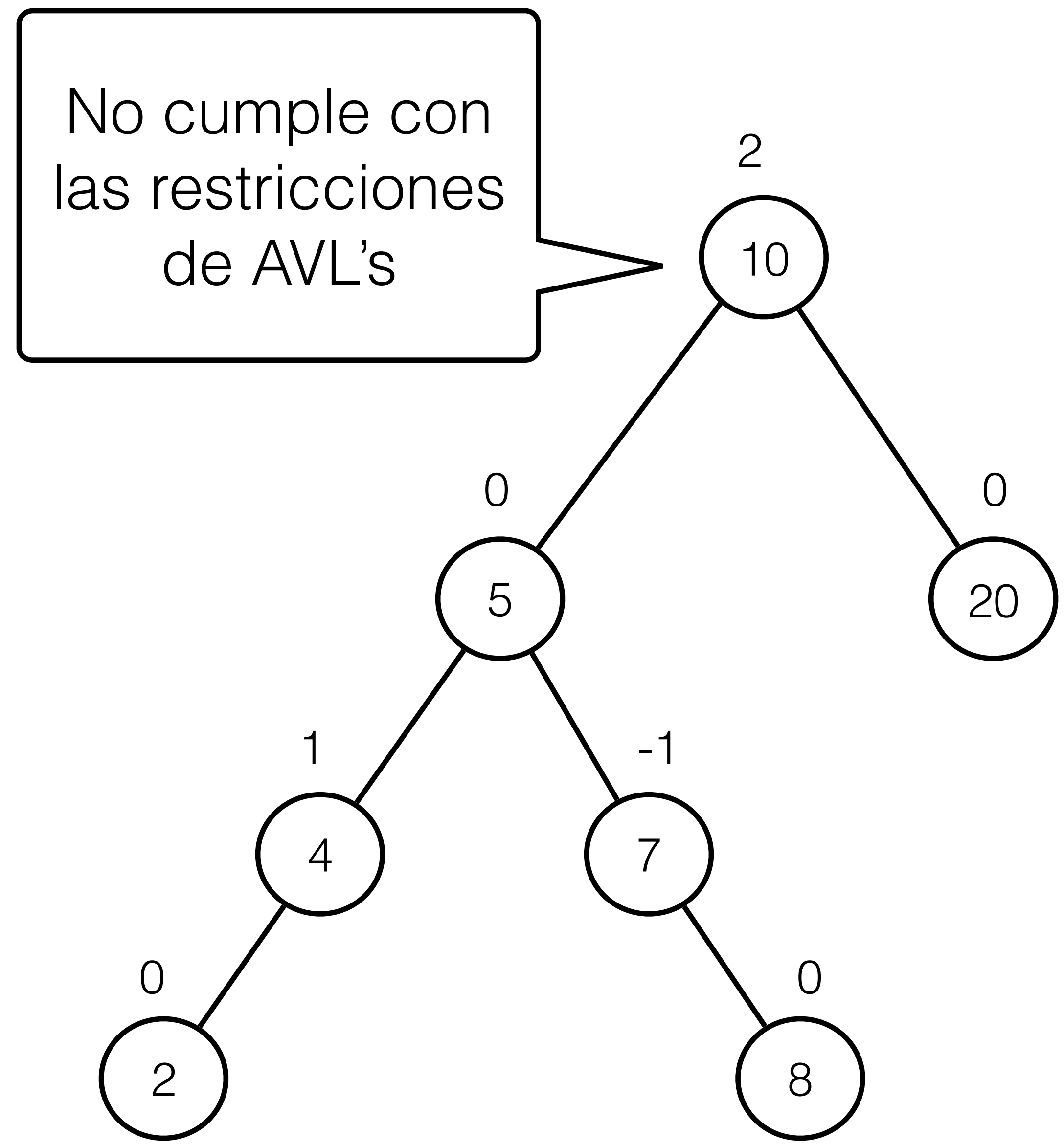
- Cada nodo lleva su factor de balance: la diferencia entre la altura del hijo izquierdo y el hijo derecho,
- El factor de balance se calcula como:

$$FB = alt(hi) - alt(hd) \; (= -1,0,1)$$

- Si el valor es -1 el hijo der. tiene más altura, si es 0 los dos tienen más altura, si es 1 el hijo izq. tiene más altura

Ejemplos


1
10
0
1
5
20
1
-1
0
4
7
12
0
0
2
8


No cumple con las restricciones de AVL's
2
10
0
0
5
20
1
-1
4
7
0
0
2
8

Altura de AVLs

Tenemos el siguiente teorema:

La altura de cualquier AVL es O(log n), donde n es la cantidad de nodos

Dem.

$$N_h = \text{cant. míntima de nodos que puede tener un AVL de altura } h$$

$$N_0 = 0$$
$$N_1 = 1$$
$$N_h = N_{h-1} + N_{h-2} + 1$$

Misma ecuación que fibonacci

entonces:

$$N_h \geq 2^{\frac{h}{2}}$$
$$\equiv$$
$$log_2 N_h \geq \frac{h}{2}$$
$$\Rightarrow$$
$$h \in O(log_2 N_h)$$

Búsqueda en AVL

- La búsqueda en AVL es igual que en ABB's:
  - Buscamos comparando con la raíz,
  - Si el elemento que buscamos es la raíz retornamos true,
  - Si es más chico que la raíz buscamos por la izquierda,
  - Si es más grande buscamos por la derecha,

La búsqueda es O(h), donde h es la altura, es decir es O(log n)

Inserción en AVL

La principal diferencia con la inserción en ABBs es que tenemos que preservar el balanceo.

- Procedemos como los ABB’s buscamos el lugar en donde insertar,
- Insertamos y vamos desde la hoja hasta la raíz, si todo los factores de balanceo están entre -1,0,1, se termina.
- En otro caso, si encontramos un 2 o -2 se debe rebalancear el árbol.

Rotaciones en AVL’s

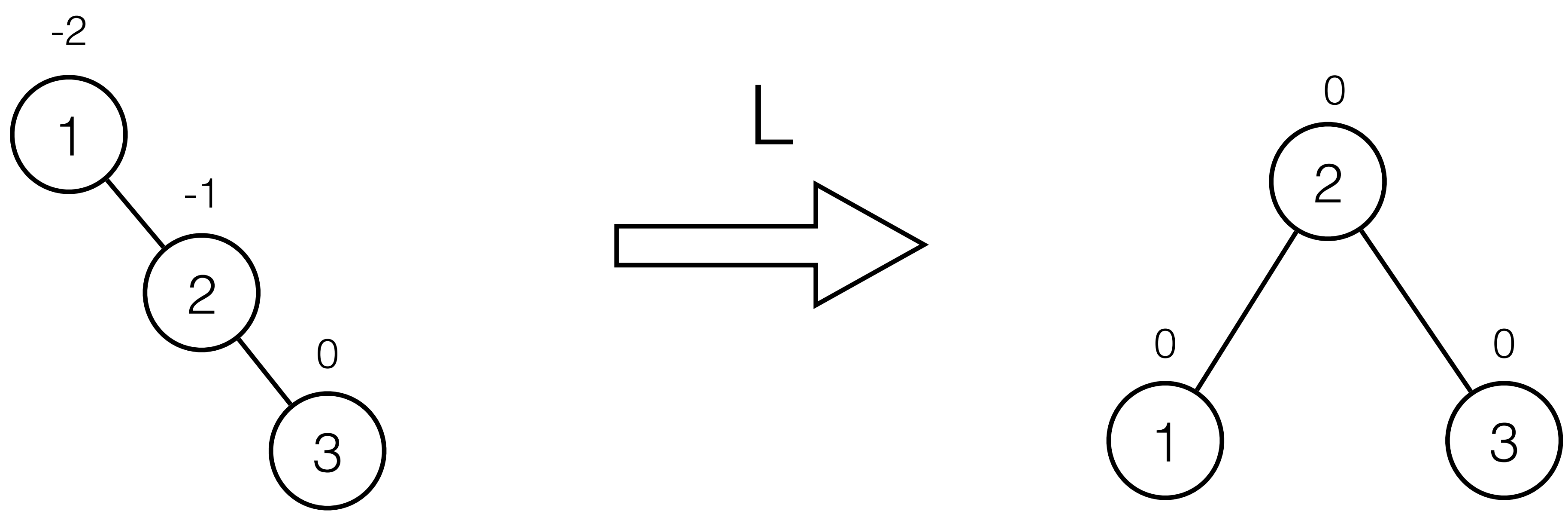
Tenemos cuatro tipos de transformaciones que permiten rebalancear un AVL:

- Rotación a la izquierda (L-rotación),
- Rotación a la derecha (R-rotación),
- Rotación izquierda-derecha (LR-rotación)
- Rotación derecha-izquierda (RL-rotación)

Estas rotaciones son simétricas.

Casos Simples

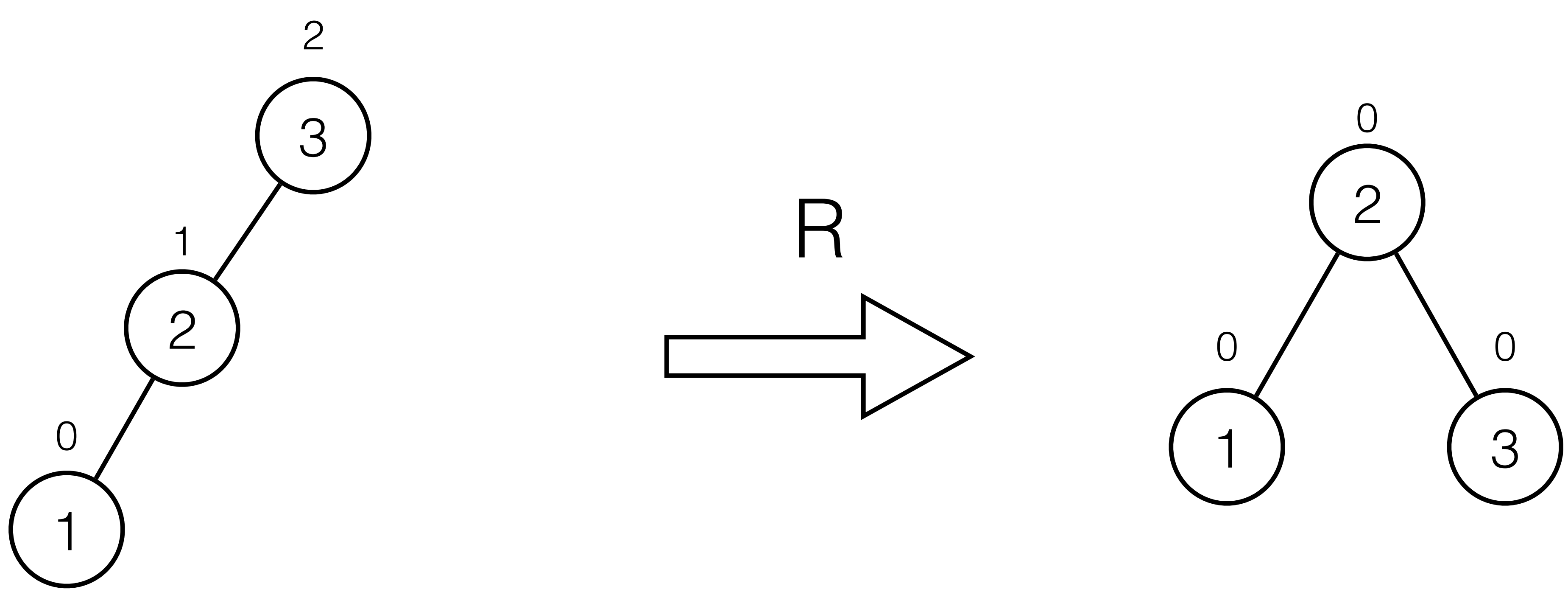
Ilustremos las rotaciones con sus casos más simples:



Notar: los factores de balanceo se acomodan, el árbol rotado tiene la misma altura que el árbol antes de la inserción

Casos Simples

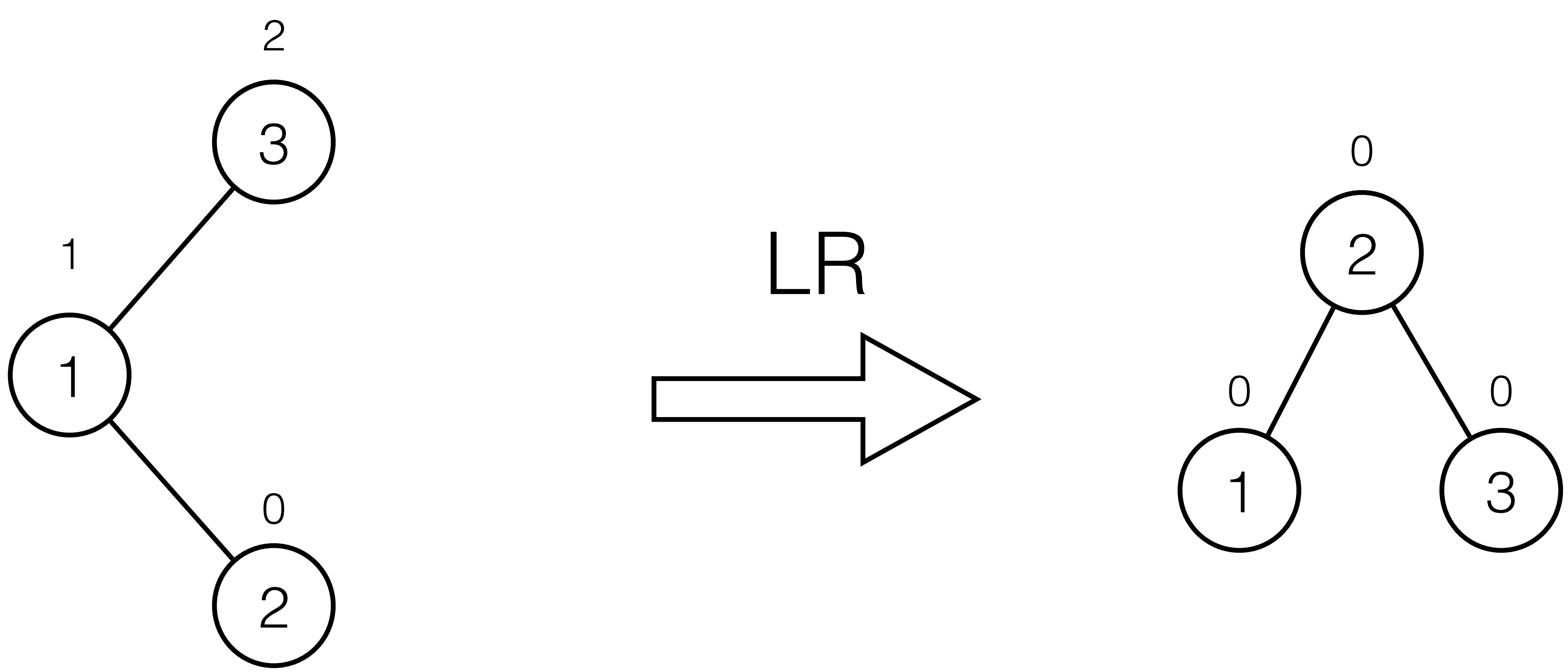
Rotación a la derecha, caso simétrico:



Notar: los factores de balanceo se acomodan, el árbol rotado tiene la misma altura que el árbol antes de la inserción

Casos Simples

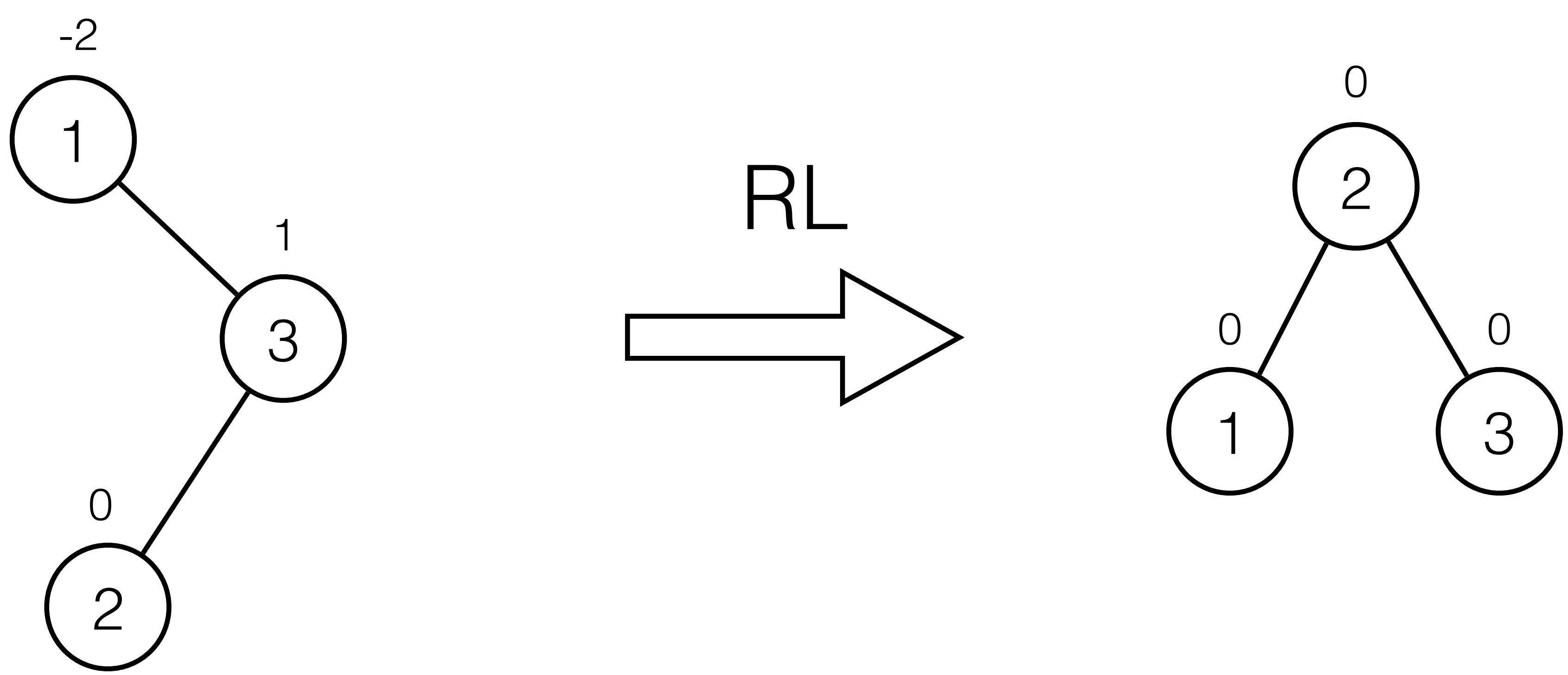
Doble rotación: Izquierda-Derecha:



Notar: los factores de balanceo se acomodan, el árbol rotado tiene la misma altura que el árbol antes de la inserción

Casos Simples

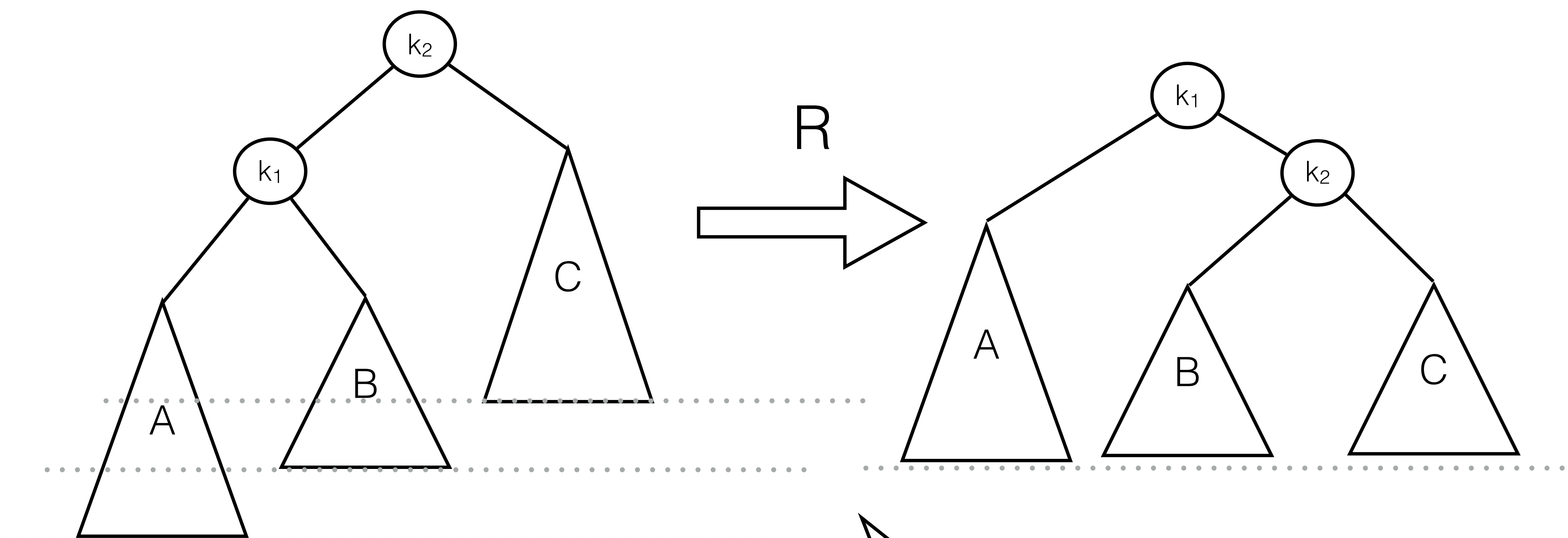
Rotación Derecha-Izquierda:



Notar: los factores de balanceo se acomodan, el árbol rotado tiene la misma altura que el árbol antes de la inserción

Caso General

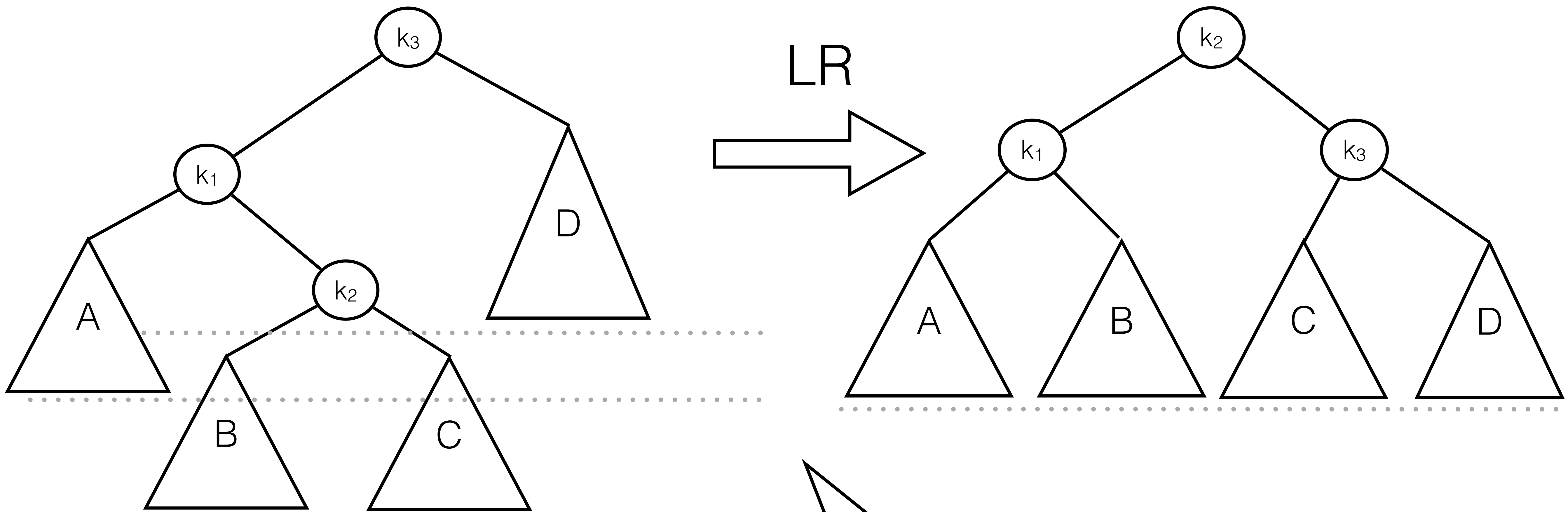
Podemos ilustrar los casos generales de la siguiente forma:



La rotación a la izquierda es simétrica

Caso General

Veamos la doble rotación Izq-Der.:



La rotación Der. Izq. es simétrica

Implementación por Casos

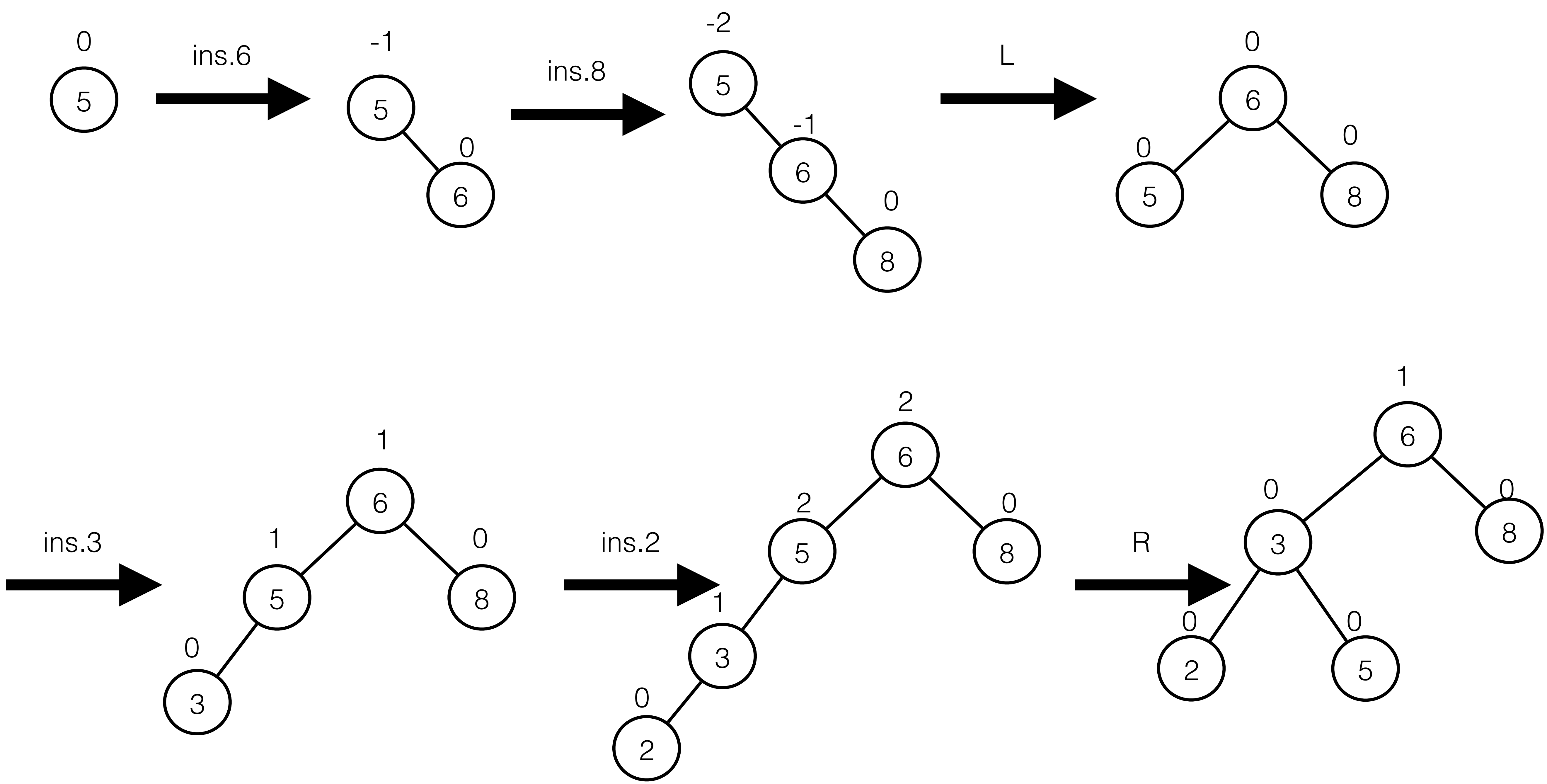
- **Caso desbalanceo der.-der.**: Si el FB de un nodo es -2, y el hijo derecho tiene FB -1 o 0. Hacemos una rotación a la izquierda.

- **Caso desbalanceo der.-izq.**: Si el FB de un nodo es -2, y el FB del hijo derecho es mayor a 0, entonces tenemos una rotación RL.

- **Caso desbalanceo izq.-izq.**: Si el FB de un nodo es 2, y el FB del hijo izq. es positivo, entonces hacemos una rotación a la derecha.

- **Caso desbalanceo izq.-der.**: Si el FB de un nodo es 2 y el FB del hijo izq. es menor que cero, entonces hacemos una rotación LR.

Un Ejemplo



0
5
ins.6
-1
5
0
6
ins.8
-2
5
-1
6
0
8
L
0
6
0
5
0
8
ins.3
1
6
1
5
0
8
0
3
ins.2
2
6
2
5
0
8
1
3
0
2
R
1
6
0
3
0
8
0
2
0
5

Borrado

- Para el borrado de un elemento se procede como en los ABB's,
- Buscamos el nodo a borrar, y reemplazamos por el nodo correspondiente,
- Vamos de abajo hacia arriba corrigiendo desbalances con rotaciones.

A lo sumo O(log n) rotaciones

Eficiencia

Debido a que los AVL’s tienen altura O(log n), todas las operaciones de diccionarios son log n, además:

- Las rotaciones toman tiempo constante debido a que son cambios de referencias,
- La implementación no esta acotada a un tamaño máximo, debido a que se emplean estructuras dinámicas,
- Las rotaciones son muy frecuentes en árboles AVL’s